Procedia Computer Science

Volume 29, 2014, Pages 1171–1181

ICCS 2014. 14th International Conference on Computational Science

Control of Artificial Swarms with DDDAS

R. Ryan McCune[1] and Greg R. Madey[1]

University of Notre Dame, Notre Dame, Indiana, United States
rmccune@nd.edu, gmadey@nd.edu

Abstract

A framework for incorporating a swarm intelligent system with the Dynamic Data Driven Application System (DDDAS) is presented. Swarm intelligent systems, or artificial swarms, self-organize into useful emergent structures that are capable of solving complex problems, but are difficult to control and predict. The DDDAS concept utilizes repeated simulations of an executing application to improve analytic and predictive capability by creating a synergistic feedback loop. Incorporating DDDAS with swarm applications can significantly improve control of the swarm. An overview of the DDDAS framework for swarm control is presented, and then demonstrated with an example swarm application.

Keywords: Swarm Intelligent Systems, Emergent Behavior, Swarm Control, Agent-Based Modeling, Design Pattern, DDDAS

1 Introduction

We are investigating methods for incorporating DDDAS for control of swarm intelligent systems. Swarm intelligent systems are artificial swarms inspired by biology, like flocks of birds and ant colonies. Swarms exhibit emergent behavior, where simple behaviors distributed across many agents give rise to a collective behavior capable solving complex problems, resulting in a whole system greater than the sum of its parts. Swarm intelligent systems are robust, scalable, adaptable, and efficient problem solvers, an attractive paradigm for multi-agent applications.

Research into the application of artificial swarms include swarm robotics and Unmanned Aerial Vehicles (UAVs) [11, 9]. UAVs are remotely piloted aircrafts that are less costly than piloted counterparts while minimizing safety risks to personnel. Application for UAV include government, military, and commercial projects. As technology improves, single UAVs will likely be supplanted by teams of UAVs for more complex objectives [2].

As coordination and communication amongst multiple UAVs becomes prohibitively complex, the application of swarm intelligence to teams of UAVs, or UAV swarms, is forseen [4]. With swarm intelligent systems, control occurs at the micro-level of the agents, while the macro-level emergent behavior meets the application objective. However, the relationship between the low-level agent behavior and the macro-level emergent behavior is non-linear, not clearly

Selection and peer-review under responsibility of the Scientific Programme Committee of ICCS 2014

doi:10.1016/j.procs.2014.05.105

understood, and difficult to predict. For a real-time swarm application such as a UAV swarm, additional tools and methods must be explored to improve control of the swarm [7, 6].

A framework is presented to incorporate a swarm application into a Dynamic Data Driven Application System (DDDAS). DDDAS entails the ability to incorporate additional data into an executing application [3]. The data helps drive the decision-making process, which in turn impacts the measurement of future real-time data, affecting future simulations. This synergistic feedback control loop between the simulations and an executing application can tremendously improve analytic and predictive capabilities of the real-time system. Developing a swarm application with DDDAS concepts can alleviate control problems arising from the non-linearity between the agent-level control behavior and application-level emergent behavior.

For a swarm application to be incorporated into DDDAS, the application must adhere to a specific swarm application architectural pattern, where performance of the swarm is measured by a single, aggregate statistic. The swarm application architecture as well as the DDDAS framework for swarm control will be discussed in Section 2. In Section 3, both the architectural pattern and the DDDAS framework will be applied to an existing swarm application. Conclusions and future work are discussed in Section 4.

2 DDDAS Framework for Swarm Control

A framework for controlling a swarm intelligent application with DDDAS is introduced. A swarm intelligent application utilizes emergent behavior of the swarm to accomplish a task, but is controlled by low-level parameters at the agent level. The relationship between agent-level behaviors and emergent behaviors is not clearly understood. This non-linear relationship between agent-level behaviors and emergent behaviors makes a swarm difficult to control, especially under dynamic conditions. Incorporating DDDAS can mitigate these challenges.

Two application designs are presented in order to incorporate DDDAS for swarm control. The first design is a swarm application architecture. For a swarm application to be integrated with DDDAS, the application must be designed to report application performance as a single, aggregated statistic. Representing swarm performance as an aggregate statistic will allow for the application to be optimized via simulation.

The second design is that of the DDDAS framework, which will incorporate the swarm application architecture. The framework will execute several simulations in parallel that utilize real-time data. The real-time data will help the simulations best reflect current conditions, and will facilitate well-informed decisions by central control. Design of the swarm application architecture is presented in Section 2.1 and the DDDAS framework is presented in Section 2.2.

2.1 Swarm Application Architecture

A specific architecture is required of the swarm application in order to be integrated within a DDDAS framework. In the swarm application architecture, one or few swarm parameters are used to control the swarm. The control parameters affect how the swarm self-organizes into the emergent behavior. The emergent behavior of the swarm is utilized to accomplish the given task. The performance of the swarm in relation to the given task is then reported back to the central controller as a single, aggregated statistic. The swarm application architecture is outlined in Figure 1.

The control of several agents by broadcasting one or few swarm parameters is an intuitive method for commanding an artificial swarm. The presented architecture is unique, however,

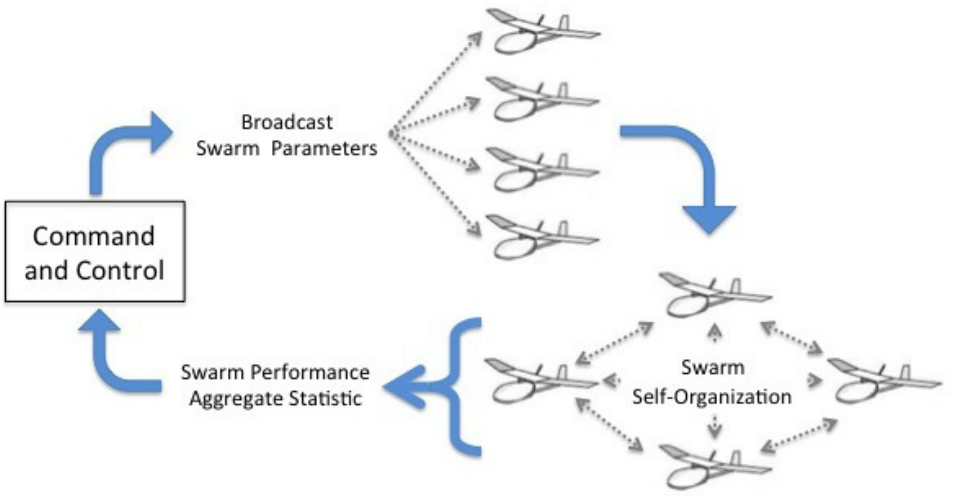

Figure 1: The swarm application architecture required for swarm control with DDDAS. The application is controlled by one or few parameters broadcast to agents, the parameters affect how the swarm self-organizes into the emergent structure, and the performance of the swarm is reported back to the central controller in a single, aggregate statistic.

for the reporting of swarm performance as a single aggregate statistic. The aggregate statistic computation is performed amongst the swarm, or another decentralized mechanism, and transmitted back to central control by a single agent. The precise nature of the statistic is at the discretion of the designer, and depends on the task and the emergent behavior. A single swarm performance statistic is in contrast to each individual agent reporting directly to central control. The aggregate swarm statistic is necessary for DDDAS integration.

2.1.1 Performance Statistics: One vs. Many

Reporting swarm performance as a single, aggregate statistic, instead of data from many agents, is favorable for several reasons. First, a single central controller may have difficulty processing disjoint information from multiple sources when trying to control a swarm. Condensing swarm reporting to an aggregate statistic simplifies processing for the central controller. Secondly, all agents repeatedly reporting data to central control, such as location, power, sensor readings, etc., would require excessive bandwidth and stress communication constraints. Moreover, repeated data transmission by mobile agents would deplete power, where battery life is of primary concern in mobile sensor networks. Lastly, the purpose of utilizing swarm behavior is for agents to self-organize in a decentralized manner. Requiring each agent to report back to central control fails to take advantage of decentralization.

2.1.2 Swarm Application Architecture Example with BOIDS

An example of the presented swarm application architeture is provided with BOIDS flocking behavior to better illustrate the design. In the BOIDS model [10], agents are controlled by swarm parameters that include the radius for detecting neighbors, as well as the relative weighting co-

efficients for each of the three flocking behaviors; namely, how much alignment, cohesion, and separation each impact the agent flight vector. An increase in the weighting of the separation parameter, for instance, will result in a flock of agents spaced farther apart, while an increase in the cohesion weighting will bring agents of the flock closer together. Increasing the neighbor visibility radius may result in faster convergence.

Once the parameters are assigned, BOIDS agents interact and self-organize into a flock. The performance of the flock must then be reported back to the central controller as a single aggregate statistic. For BOIDS, the aggregate statistic could be the density of the flock, total coverage, number of agents in the flock, etc. The precise statistic depends on the application and is left to the discretion of the designer.

2.2 DDDAS Swarm Control Framework

A DDDAS framework is presented for command and control of a swarm application. The framework incorporates the swarm application architecture. While the swarm application architecture offers advantages, the relationship between agent behaviors and emergent behavior remains nebulous. How a central controller may best control agent parameters under dynamic conditions is not immediately clear, as even minor adjustments could drastically impact emergent behavior. Utilizing the swarm application architecture, the DDDAS framework facilitates improved analytic, predictive, and decision-making capabilities for swarm applications.

The DDDAS framework augments the swarm application with simulations. At a given point during the swarm application, many simulations are executed in parallel. All the simulations utilize real-time data to accurately represent current conditions, but each simulation incorporates different agent-level control parameters. Because of the swarm application architecture, each simulation outputs the swarm performance as an aggregate statistic. The results of each simulation, the swarm performance metric, can be plotted against the underlying agent-level parameters, allowing a central controller to best determine how to control the swarm under current conditions. Adjustments to agent-level behavior will impact the measurement of future agent data, which in turn affect real-time data for the next simulations, achieving the DDDAS feedback control loop. The DDDAS framework is diagrammed in Figure 2.

The feedback control loop depicted in Figure 2 occurs repeatedly throughout the duration of the application. The precise time frame for execution of each iteration is at the discretion of the system designer. For example, the set of parallel simulations may be executed at a fixed time step, on command from the central controller, or on automatic detection of an event.

2.2.1 The Simulations

In the presented DDDAS framework, several simulations are executed in parallel. The simulations are agent-based simulations, which are well-suited for modeling swarms.

The simulations are initialized with two sets of inputs:

- Real-time data from the application
- Swarm control parameters for the simulated agents

All simulations executed at the same time receive the same real-time data. Real-time data includes all information necessary to represent the current conditions of the application in a simulation. A simulation also receives a value for the swarm control parameter. The value for the swarm control parameter varies across the many simulations,

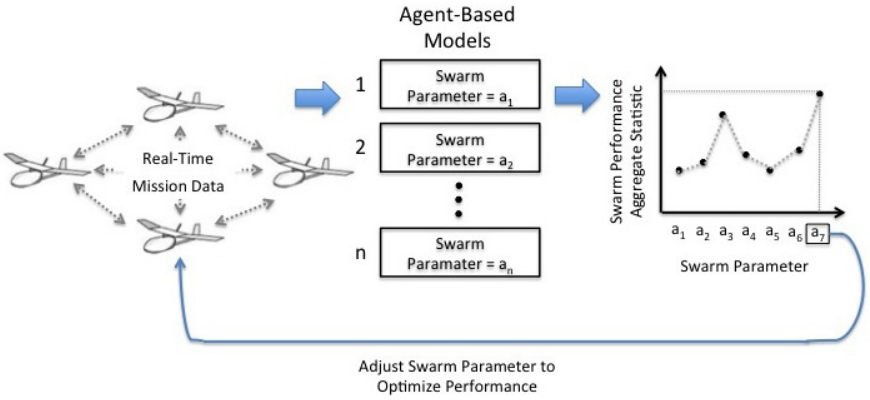

Figure 2: Diagram of the DDDAS framework with feedback control loop for swarm control, where real-time application data is inputted into agent-based simulations, and simulation results drive adjustment to application parameters. The simulations execute across different agent-level parameters, and when plotted against the resulting swarm performance metric, allows the controller to appropriately adjust agent parameters for the application.

Enough simulations are employed to cover a reasonable spectrum of the swarm control parameter space, to ensure all possible scenarios are simulated. Duplicate simulations may be run with the same swarm control parameter to account for randomness in the simulations.

2.2.2 Real-Time Data

Real-time data is utilized by simulations to represent current conditions of agents and the environment. The real-time application data is obtained from transmission by each agent. This does not contradict the earlier stated advantages of decentralized control and the swarm application architecture from Section 2.1.1, where a single, aggregate statistic preserves power and bandwidth.

While agents do transmit data back to centralized entity to run the simulations, the data is only transmitted intermittently, at an interval equal to the time in between sets of simulation executions. Moreover, not all data from agent sensors needs to be transmitted, only the amount required by the simulation to appropriately represent the current application environment. So while some data needs to be transmitted by each agent for the DDDAS framework to operate, the information sent and power required for such transmissions is much less than if all agents were continuously reporting all sensor data. Otherwise, agents are largely autonomous and still only receive agent-level swarming parameters from a remote operator, preserving decentralized control.

3 Decentralized Clustering Application with DDDAS

With an introduction to the DDDAS framework for swarm control, the framework will be applied to a swarm application. The swarm application is Decentralized K-medians clustering, previously introduced in [8]. Based on the foraging behavior of ants, an artificial swarm is created that self-organizes into clusters. First, Decentralized K-medians clustering will be introduced, then the DDDAS framework will be applied.

3.1 Decentralized K-medians Clustering

Decentralized K-medians clustering is a technique for an artificial swarm to identify cluster centroids that approximates k-medians clustering. In the *k-medians* problem, given a set S of n points in a metric space and a positive integer k, the task is identify k *medians* in the space, such that the cumulative distance between each point in S and the nearest median is minimized. The problem has been proven NP-hard and numerous approximation algorithms have been developed [5, 1].

The traditional form of this problem is centralized, where all information is available for centralized computation. For the decentralized version of the problem, a multi-agent system is presented where certain agents self-organize to the k medians. The decentralized clustering problem is applied to an application scenario incorporating UAVs.

3.1.1 Scenario

Picture a scenario where there are n small fires ablaze in a large field. In order to extinguish the fires, several UAVs are employed, which are equipped with buckets to hold water for deploying onto fires. Also available are k tankers filled with water. Each UAV is designed to fill up its bucket with water from a tanker, locate a fire, deploy the payload, then return to a tanker for refill. In order to efficiently extinguish all the fires, tankers should be positioned so as to minimize the cumulative distance traveled by UAVs between tankers and the fires, like the k-medians problem.

However, for some reason, whether smoke from the fires, or bandwidth concerns, or security reasons, centralized control of the UAVs or tankers is not possible. There exists limited communication between central control and UAVs, and UAVs do not communicate with tankers unless directly overhead. Global information, such as location coordinates, are not available. In order for tankers to relocate to efficient locations, the system must self-organize through swarm intelligent behaviors.

3.1.2 Swarm Intelligent Solution

The solution presented in [8] adapts ant foraging behavior. Ants utilize a stigmergic environment, where physical alterations to the environment affect the decisions of future agents. For the above scenario, UAVs utilize digital pheromones. UAVs deposit digital pheromones in the environment while randomly searching for fires. Once a fire is located and the water is deployed, UAVs return to a tanker by following the path of highest pheromone concentration. By the nature of the pheromone deposits, the path will always lead back to a tanker. Upon returning, a tanker senses the returning direction of the UAV, and moves one grid unit in that direction. An overview of the process is depicted in Figure 3.

Over time, the cumulative distance between the fires and nearest tanker reduces. An agent-based simulation was developed to evaluate the technique, with the global minimum achieved in several instances. For further implementation details and results, refer to [8].

3.2 Swarm Application with DDDAS

The Decentralized K-medians clustering application is incorporated with a DDDAS framework. Implementing DDDAS for the decentralized clustering application can help optimize swarm performance, like providing the best chance for the swarm to reduce the cumulative distance between medians (tankers) and data points (fires). We will assume the terminology presented in Section 3.1.1 of tankers, fires, and UAVs for the artificial swarm. The swarm application

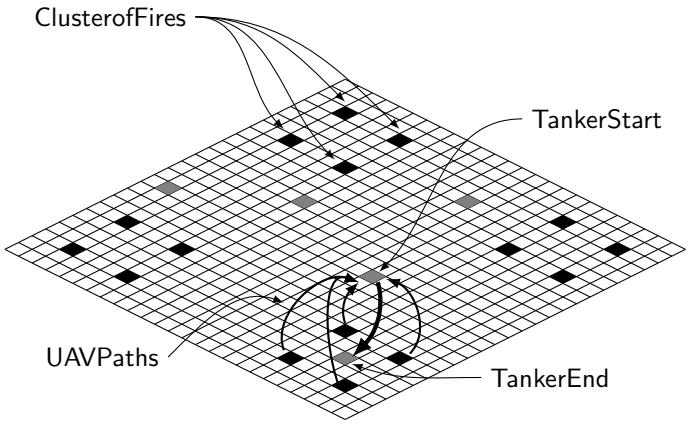

Figure 3: Scenario depicting fires and tankers. Fires are black grids arranged in square clusters. Tankers are gray grid squares. The UAVs, not pictured, randomly search the environment from a tanker to a fire, then follow pheromones back to the tanker. The tanker moves one grid in the direction of the returning UAV. Over time, tankers relocate to locations that reduce the distance between fires and nearest fires, like in the middle of the square fire clusters at Tanker End.

will first be adapted to the swarm application architecture, then the DDDAS framework will be applied.

3.2.1 Swarm Application Architecture for Decentralized Clustering

The Decentralized K-medians clustering system is autonomous, requiring no human operation after initialization. To adapt the clustering system to the swarm application architecture, an agent control parameter needs to be implemented, and the performance of the swarm needs to be quantified by an aggregate statistic. To meet the requirements of the swarm application architecture, some communication constraints imposed by the scenario are slightly relaxed. Specifically, instead of strict autonomy, a central controller can broadcast a single parameter to all UAV agents, and the environment has the capability to transmit digital pheromone data to central control.

Control and performance parameters of the swarm will be designed around the Random Action Probability (RAP) parameter. While the system has been evaluated using the cumulative distance between fires and nearest tankers, implementing such a performance parameter would require protocols that would either be too complex or negate the need for decentralized self-organization. The RAP parameter dictates how UAVs follow the digital pheromone trail back to a tanker. UAVs follow pheromones back to a tanker, but with a certain probability will take a random action. For example, if the RAP is 0.1, then 10% of the time a UAV will follow a random direction instead of the pheromone trail. The parameter ensures that other parts of the environment are explored by the UAVs, uncovering the possible closer tanker. A properly calibrated RAP finds a balance between agents exploring the environment and agents following the established pheromone gradients. The swarm application architecture's control and performance parameters are based on the RAP parameter.

In the swarm application architecture for the decentralized clustering system, the agent

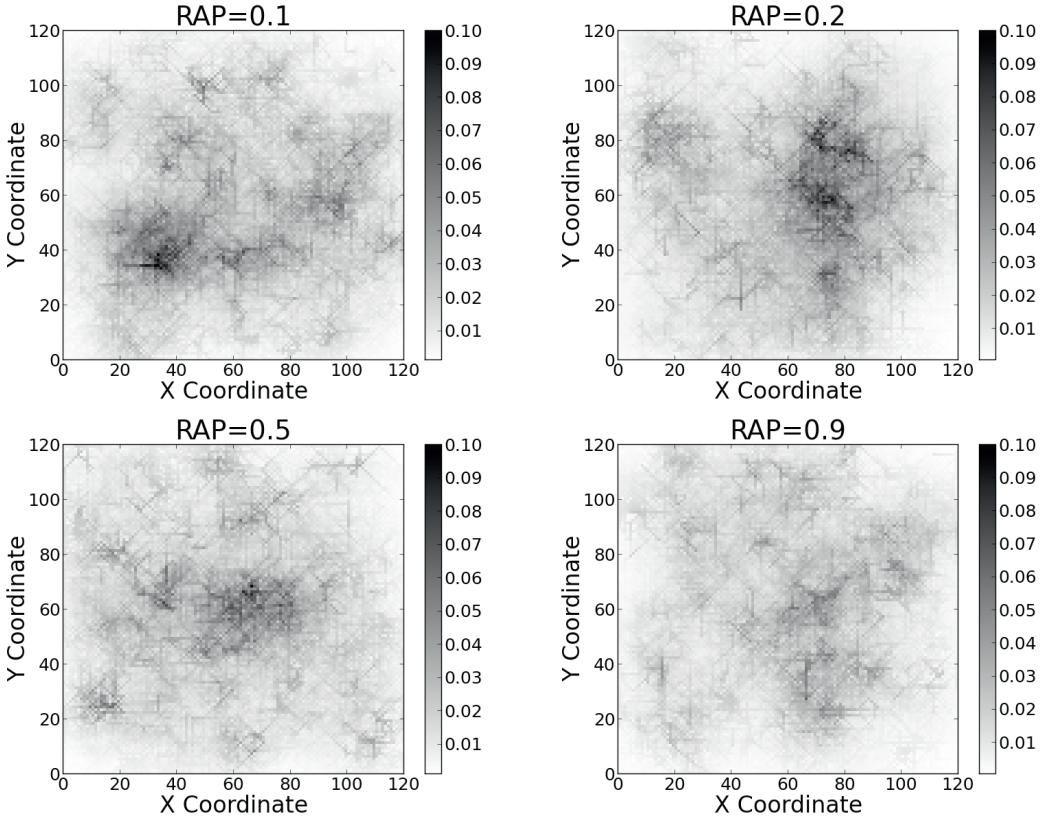

Figure 4: Color maps depicting pheromone concentrations for different Random Action Probabilities (RAP). A low RAP means UAVs travel more deterministically, and thus pheromones will be more concentrated in fewer areas. A high RAP means UAVs are more likely to explore, and pheromones will be more evenly distributed throughout the environment. Notice that an RAP=0.2 yields more areas of higher concentration, as agents explore more than RAP=0.1 but still locate tankers. In RAP=0.5 and RAP=0.9 pheromone concentrations are less dense and more distributed as UAVs are more likely to explore the environment than return to tankers.

control parameter is the RAP parameter. A central controller can adjust the RAP to affect how often a UAV will explore an alternative path to the pheromone gradient back to a tanker. For the swarm performance parameter, the variance of the environment's pheromone concentration is returned to central control. A larger variance means a greater distribution of pheromones in the environment, and is used as a heuristic for UAVs finding good paths to nearby tankers.

Agent-based simulations were run to demonstrate the impact of RAP on pheromone concentration. Agent-based simulations were run 100 times for 10,000 time steps in a 120 by 120 non-toroidal grid environment, with 16 fires randomly placed throughout the environment. Figure 4 shows color maps of each environment's average pheromone distribution for RAP of 0.1, 0.2, 0.5, 0.9, depicted for illustrative purposes. The pheromone concentration variance is provided in Table 1

RAP	Variance
0.1	1.54e-3
0.2	1.96e-2
0.5	1.21e-4
0.9	8.1e-5

Table 1: Standard deviation measures for pheromone concentrations depicted in Figure 4

3.3 DDDAS Framework for Decentralized Clustering

The decentralized clustering system is incorporated with a DDDAS framework. Real-time application data is used to initialize the simulations. The RAP parameter is varied between simulations while the rest of the input data remains consistent. Each simulation outputs a swarm performance statistic, the variance of the pheromone concentration. The variance is plotted against the RAP, allowing the central controller to make an optimal decision under current conditions. The process is depicted in Figure 5.

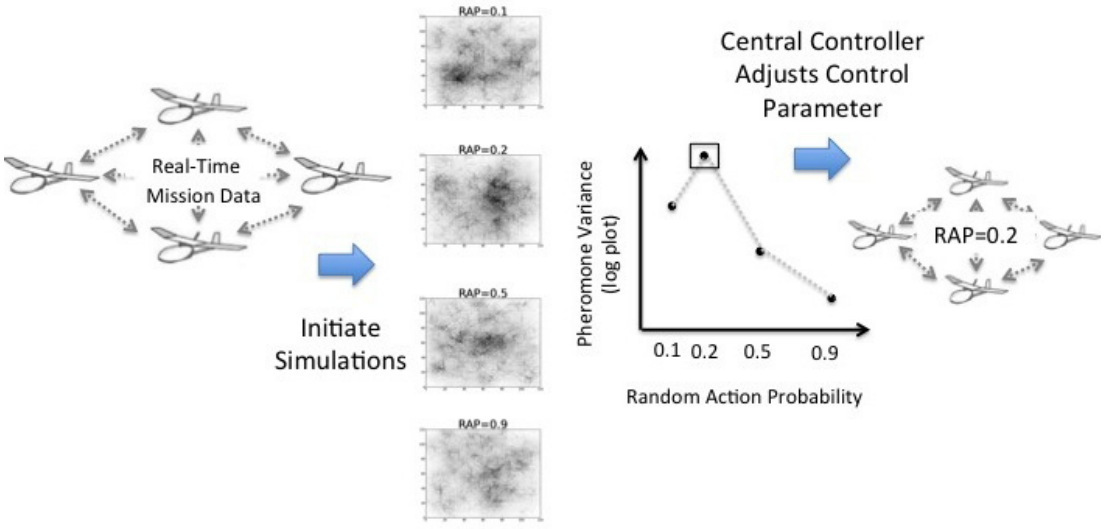

Figure 5: Diagram of the DDDAS framework applied to Decentralized K-medians clustering. Simulations utilizing real-time application data output swarm performance as a single, aggregate statistic, allowing central control to appropriately adjust swarm control parameters.

Some constraints of the original problem are relaxed to make the simulations possible. In the original problem, the swarm is decentralized, and a remote operator is not aware of fire, tanker, or UAV locations. The simulations are provided real-time data to simulate current conditions. If the real-time data means locations of fires, tankers, and UAVs, then a central controller could potentially calculate the distance between fires and tankers. Still, agents may be able to broadcast data back to central control, but a remote operator cannot control individual agents, just the swarm parameter that is broadcast to all agents. Agents still self-organize, so the presented scenario is acceptable for illustrative purposes. Further work for DDDAS and swarm

control includes resolving the trade offs between self-organization and centralized control.

4 Conclusions and Future Work

A DDDAS framework for swarm control was presented. The design improves analytic and predictive capabilities of the swarm controller by executing many parallel simulations utilizing real-time data from the swarm application. Performance of the swarm application is captured by a single aggregate statistic, allowing for swarm control parameters to be optimized via simulation. The framework incorporates agent-based simulations of the environment and provides a powerful tool for swarm operators.

Further work includes exploring different time frames for implementing simulations with a real-time application. Time frames include how long, or how far into the future, a simulation should run, and how such effectiveness can be evaluated. For example, a simulation running too short may not capture important behavior, but a simulation running too long may quickly diverge from observed behavior and no longer be useful. Time frame exploration can also explore the interval between simulation executions, and the trade-off between the interval between simulations, and the length of the simulation.

Inherent to swarm control is a trade-off between centralized control and decentralized self-organization. Self-organization offers many advantages, including avoidance of communication bottlenecks that plague centralized control of many agents, so retaining control of a self-organizing system is a contradiction, albeit necessary. Incorporating DDDAS for swarm control requires simulations. In the presented framework, real-time data is provided to simulations that is not necessarily available to central control. Questions for future work include how simulations can be incorporated without requiring additional information of the system for central control.

Acknowledgements

This work was funded in part by the Air Force Office of Scientific Research under the DDDAS program No. FA9550-11-1-0351, as well as the Department of Education's GAANN Fellowship provided through the University of Notre Dame's Computer Science Department.

References

[1] Sanjeev Arora, Prabhakar Raghavan, and Satish Rao. Approximation schemes for euclidean k-medians and related problems. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, pages 106–113. ACM, 1998.

[2] Stephen A Cambone. *Unmanned aircraft systems roadmap 2005-2030*. Defense Technical Information Center, 2005.

[3] Frederica Darema. Dynamic data driven applications systems: A new paradigm for application simulations and measurements. In *Computational Science-ICCS 2004*, pages 662–669. Springer, 2004.

[4] Paolo Gaudiano, Eric Bonabeau, and Ben Shargel. Evolving behaviors for a swarm of unmanned air vehicles. In *Swarm Intelligence Symposium*, pages 317–324. IEEE, 2005.

[5] Anil K Jain, M Narasimha Murty, and Patrick J Flynn. Data clustering: a review. *ACM computing surveys (CSUR)*, 31(3):264–323, 1999.

[6] Gregory R Madey, M Brian Blake, Christian Poellabauer, Hongsheng Lu, R Ryan McCune, and Yi Wei. Applying dddas principles to command, control and mission planning for uav swarms. *Procedia Computer Science*, 9:1177–1186, 2012.

[7] R Ryan McCune and Gregory R Madey. Swarm control of uavs for cooperative hunting with dddas. *Procedia Computer Science*, 18:2537–2544, 2013.

[8] R Ryan McCune and Gregory R Madey. Decentralized k-means clustering with manet swarms. In *Proceedings of the Agent-Directed Simulation Symposium*. Society for Computer Simulation International, 2014. To appear.

[9] Ryan McCune, Rachael Purta, Mikolaj Dobski, Artur Jaworski, Greg Madey, Alexander Madey, Yi Wei, and M Brian Blake. Investigations of dddas for command and control of uav swarms with agent-based modeling. In *Simulation Conference (WSC), 2013 Winter*, pages 1467–1478. IEEE, 2013.

[10] Craig W Reynolds. Steering behaviors for autonomous characters. In *Game Developers Conference Proceedings*, pages 763–782, 1999.

[11] Erol Şahin. Swarm robotics: From sources of inspiration to domains of application. In *Swarm robotics*, pages 10–20. Springer, 2005.